Surveying Some Metaheuristic Algorithms For Solving Maximum Clique Graph Problem

Quoc Phan Tan
Faculty of Information Technology, Saigon University, VietNam

Correspondence: quocpt@sgu.edu.vn
Communication: received 30 Aug 2022, revised 02 Feb 2023, accepted 12 Feb 2023
DOI: 10.32913/mic-ict-research.v2023.n1.1197

***Abstract*: Maximum clique graph problem is a combinatorial optimization that applies science and engineering such as social networks, telecommunication networks, bioinformatics, etc. Maximum clique is a problem of class NP-hard. There are many approaches for solving the maximum clique graph problem such as algorithms to find the exact solutions, heuristic algorithms, metaheuristic algorithms, etc. In this paper, we survey the approach for solving the maximum clique graph problem in the direction of metaheuristic algorithms and evaluate the quality of these research based on the experimental data system DIMACS. This survey can be useful for further research on maximum clique graph problems.**

Keywords: *maximum clique problem, social networks, heuristic algorithm, metaheuristic algorithm, DIMACS, soft computing.*

I. INTRODUCTION

1. Definitions

This section presents some basic definitions for the maximum clique problem [42, 44].

Definition 1: Clique

Given a connected undirected graph $G = (V, E)$; where V is the vertex set and E is the edge set. The vertex set $C \subseteq V$ is said to be a clique of the graph G if every pair of vertices (u, v) in C is an edge in the set E; i.e. vertices belonging to the clique will be the vertices of a complete graph.

The number of vertices (also called the size) of a clique is denoted by $|C|$. If a clique C contains vertex v then $|C| \leq deg(v) + 1$; where $deg(v)$ is defined as the degree of vertex v.

Definition 2: Maximal Clique

C is said to be a maximal clique of a graph G if C is a clique and C does not belong to any other clique of a graph G that has more vertices than it does.

Definition 3: Maximum clique

C is said to be a maximum clique of a graph G if C is a clique and C has the most vertices among the cliques of G. The number of vertices of maximum cliques in a graph G is denoted by $\omega(G)$ and it is called the clique index of the graph G.

A maximum clique is a maximal clique; however a maximal clique is not necessary in a maximum clique. A graph can have multiple cliques of the same size as maximum cliques.

Definition 4: The maximum clique problem

Given a connected undirected graph $G = (V, E)$; where V is the vertex set and E is the edge set. The problem of finding maximum clique (Maximum Clique Problem - MCP) is the problem of finding a maximum clique in a given graph. In the general case, the proven maximum clique problem belongs to the NP-*hard* class [23, 44].

If $G = (V, E)$ is an unweighted graph, then $\omega(G)$ =max$\{|C|$: C is a clique of the graph $G\}$. If $G = (V, E)$ is a weighted graph (on vertex), then the size of the largest clique C (clique weight) of G denoted $w(C)$ is equal to the sum of the weights vertices of C; then $w(G)$ =max$\{|C|$: C is a clique of the graph $G\}$ [11, 37, 39]. In the scope of this paper, we only consider the maximum clique problem in the case of unweighted connected undirected graphs.

Example:

Given a connected undirected graph with 9 vertices and 28 edges shown in Figure 1; a maximum clique found for this graph is the vertices {1,2,4,5,7,8}.

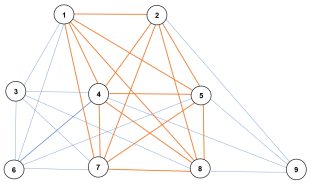

Figure 1. Graph illustration and a maximum clique with vertices {1,2,4,5,7,8}

2. Application of maximum clique problem

The MCP problem can be applied in a number of the fields of science and engineering; such as social networks, telecommunications networks, bioinformatics, scheduling [2, 8, 18, 36, 48].

For example, in a social network: each member can connect with one or other members (the connection criteria could be the same interests, for example). The problem is to find a community which has most members (or to find a community with k-members) and each member in this has a (direct) connection to all other members.

In networks: Finding and collecting information about calls, call frequency, groups of callers with high association is important because it is possible to evaluate customer patterns and optimize system performance.

In bioinformatics: The maximum clique algorithm is used to find the community for biological networks, the problem of chemical molecular structure matching.

3. Some studies related to the MCP problem

The MCP problem has been interested by many research groups [1, 12, 13, 21, 22, 25, 34, 51, 52]; These studies can be divided into three approaches:

The first direction is algorithms to find the exact solution with branching algorithms [3, 19, 23, 27], dynamic programming algorithms [33, 44]. The advantage is to find the exact solution. However, it can only solve problems with small graphs. In addition, this approach is also an important basis for evaluating the accuracy of approximation algorithms. Correctly solving the MCP problem is a big challenge in combinatorial optimization theory [8, 42, 44].

The second direction is heuristic algorithms. Heuristic algorithms refer to specific experiences for finding a solution to a particular optimization problem. The heuristic algorithm usually finds an acceptable solution in the given time, but it is not certain that it is the exact solution; even heuristic algorithms are unlikely to be efficient on all data types for a particular problem. The outstanding advantage of heuristic algorithms in solving MCP problems is the fast running time [6, 14, 16, 18, 36, 42, 50]. The result of the heuristic algorithm can be used as the branching condition in the branching algorithms.

The third direction is metaheuristic algorithms. The metaheuristic algorithm uses many heuristics combined with auxiliary techniques to explore the search space; The term "metaheuristic" often refers to individual algorithms such as local search algorithm, simulated annealing algorithm, tabu search algorithm, etc. or population algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), bee swarm algorithm, bat swarm algorithm, etc. Metaheuristic algorithm belongs to the class of optimal search algorithms. Some typical publications solving the MCP problem in this direction are the local search algorithm [10, 15, 20, 41, 43], the tabu search algorithm [33], the variable neighborhood search algorithm. [32], genetic algorithm [7, 33, 40], bee swarm algorithm [9, 45], ant colony optimization algorithm [24, 38], etc. Until now, metaheuristic approach to solving the MCP problem gives the best solution quality among the approaches to approximate solving the MCP problem.

In this paper, we will examine the approach to solving the maximum clique problem mainly in the direction of metaheuristic algorithms. We evaluate the quality of these studies based on standard experimental data system DIMACS [4].

II. METAHEURISTIC ALGORITHM SOLUTION OF OPTIMIZATION PROBLEM

1. General diagram of metaheuristic algorithms to solve optimization problems

General diagram of metaheuristic algorithms to solve optimization problems includes the following steps [30, 46, 47].

```
1  Initialize the initial solution (the solution can be an
     individual/or a population);
2  Define reinforcement strategies;
3  Define diversification strategies;
4  Calculate the fitness for each solution;
5  while (stopping condition is not satisfied)
6  {
7  Implement reinforcement strategies;
8  Implement diversification strategies;
9  Update best solution;
10 }
11 Return the best solution found;
```

2. Initialize the original solution

In metaheuristic algorithms, the initial solution is initialized randomly or according to one or several individual heuristics of the problem. Generally, the initial solution is randomly initialized to increase the diversity of the solution instances.

3. Solution intensification Strategy

The primary purpose of intensifiction is to explore deeper regions of the potential solution space in the hope of finding better quality solutions. The intensification strategy is usually implemented by applying the search strategy towards the best neighbor. To be more specific, if the intensification is too high, it can cause premature convergence problems;

local optimal solutions are found to be highly skewed or meaningless; conversely, if the intensification is too low, convergence will be slow.

Solution Intensification is often performed based on neighborhood search strategies. Depending on the characteristics of the problem, different neighborhood search methods can be applied. The intensification strategy is used throughout the course of the algorithm.

4. Solution diversification strategy

Regarding the purpose of diversifiction, it is to explore new regions of the search space to avoid the search falling into the local optimization trap. It is usually accomplished through the use of randomness and can be combined with a number of heuristics specific to each problem. Particularly, if the diversification is too high, there will be many regions of the solution space that are randomly mined, thereby slowing down the convergence of the algorithm; On the contrary, if the diversity is too low, the space of solutions to be explored will be limited, therefore the solutions found tend to converge locally or make no sense.

When a solution is given successive neighborhood searches in some ways, after a number of iterations, the quality of the solution will no longer improve. If you continue to search for neighbors in that direction, it will not be possible to upgrade the quality of the solution. Diversification strategy is often used when the quality of the existing solution is not improved after a specified number of iterations.

A metaheuristic algorithm is considered efficient if it is capable of generating a variety of solutions for efficient mining of the search space. At the same time, it also needs to be able to effectively enhance the neighborhood search around good solutions. Reinforcement and diversity are two somewhat opposing components, utilizing the combination between them is the key to the success of any metaheuristic algorithm [28–30, 46, 47].

5. Stop conditions of metaheuristic algorithms

There are many stopping conditions that have been applied in metaheuristic algorithms. Normally, three of them are used: first, the best found solution of the problem does not improve after a predetermined number of iterations; second, the number of iterations of the algorithm reaches a predetermined value; third, the algorithm runs for a predetermined amount of time (these parameters are suggested from the parameter experiment).

6. Criteria for evaluating the quality of metaheuristic algorithms

Metaheuristic algorithms are often evaluated based on the quality of the solution (the goodness of the solution) and the corresponding computational time to get that solution through experimenting on the standard datasets (benchmarks) of the problem.

The criterion for evaluating the quality of the solution is recorded as the best value (and possibly the mean and the standard deviation value) after a number of runs for each data set.

The time criterion can be assessed indirectly through the number of solutions examined. Currently, the work of Jack J. Dongarra [17] is often used to compare the computation time of algorithms on computers with different configurations. This work uses the unit of measure Mflop/s (Million Floating-point Operations Per Second – million floating point operations per second) to measure the performance of computers. The results of the study are tables of speed data measured in Mflop/s for each computer configuration. Using these dashboards we don't need to reinstall someone else's algorithm. Comparative computation time of algorithms on computers with different configurations is approximate; However, these time comparison tables provide an essential reference view of the computation times of the algorithms.

III. SURVEYING SOME METAHEURISTIC ALGORITHMS TO SOLUTION MAXIMUM CLIQUE

1. Experimental data

We chose to investigate algorithms to solve the MCP problem with 37 datasets in the DIMACS experimental data system [4]; as detailed in Table I. In DIMACS graphs have the number of vertices in the range 125 to 4000 and the number of edges in the range 6963 to 5506380. The MCP problem was a big challenge for the research teams. The goal of studying the approximate solution of the MCP problem is to improve the quality of the solution and the running time corresponding to the quality of that solution.

Table I has the following structure: The first column (Instance) is the names of the data sets in the standard experimental data system DIMACS, the next columns NODES, EDGES, DEGREE_{min}, DEGREE_{max} and Best known record the information respectively: the number of vertices, the number of edges, the number of minimum degrees, the maximum number of degrees of the vertices of the graph, the best known clique size of the graph (this is the best record updated on the experimental record update websites for the MCP problem [4, 5] to 2018). For data sets where the found record is already the optimal solution,

TABLE I
EXPERIMENTAL DATA SETS IN DIMACS SYSTEM

Instance	NODES	EDGES	DEGREE$_{min}$	DEGREE$_{max}$	Best known
C125.9	125	6963	102	119	34*
C250.9	250	27984	203	236	44*
C500.9	500	112332	431	468	57
C1000.9	1000	450079	868	925	68
C2000.9	2000	1799532	1751	1848	80
DSJC1000_5	1000	499652	447	551	15*
DSJC500_5	500	125248	220	286	13*
C2000.5	2000	999836	919	1074	16
C4000.5	4000	4000268	1895	2123	18
MANN_a27	378	70551	364	374	126*
MANN_a45	1035	533115	1012	1031	345*
MANN_a81	3321	5506380	3280	3317	1100
brock200_2	200	9876	78	114	12*
brock200_4	200	13089	112	147	17*
brock400_2	400	59786	274	328	29*
brock400_4	400	59765	275	326	33*
brock800_2	800	208166	472	566	24*
brock800_4	800	207643	481	565	26*
gen200_p0.9_44	200	17910	165	190	44*
gen200_p0.9_55	200	17910	164	190	55*
gen400_p0.9_55	400	71820	334	375	55*
gen400_p0.9_65	400	71820	333	378	65*
gen400_p0.9_75	400	71820	335	380	75*
hamming10-4	1024	434176	848	848	40*
hamming8-4	256	20864	163	163	16*
keller4	171	9435	102	124	11*
keller5	776	225990	560	638	27*
keller6	3361	4619898	2690	2952	59
p_hat300-1	300	10933	23	132	8*
p_hat300-2	300	21928	59	229	25*
p_hat300-3	300	33390	168	267	36*
p_hat700-1	700	60999	75	286	11*
p_hat700-2	700	121728	157	539	44*
p_hat700-3	700	183010	408	627	62
p_hat1500-1	1500	284923	157	614	12*
p_hat1500-2	1500	568960	335	1153	65
p_hat1500-3	1500	847244	912	1330	94

the clique size in the corresponding Best known column is marked with *.

Through the survey of the experimental data sets in Table I, we realize that: The graphs in the DIMACS system are all thick graphs [26], there is no graph with vertices of order smaller than the size of the clique found, obtained by known greedy algorithms. Therefore, the graph reduction in the direction uses a greedy algorithm to find a maximum clique; then traversing the vertices of the graph and removing vertices of order less than the size of maximum clique is not feasible.

2. Experimental results

The quality of the algorithms is noted to solve the MCP problem on the DIMACS data system as shown in Table II; in which the first column (Instance) records the names of the data sets in the DIMACS system, the next two columns Heu1, Heu2 record the record of heuristic algorithms found [31], the GEN column records the Genetic algorithm record by Elena marchiori [10], the HGAMC column records Bo Huang's algorithm [7], the ANT column records the ant swarm optimization algorithm of Serge Fenet and Christine Solnon [38], the KLS column records the algorithm's record by Kengo Katayama et al. [20], the PLS column records the algorithm record of Wayne Pullan et al. [43], the BLS column records the algorithm record of Una Benlic and Jin-Kao Hao [41], the Heu column algorithm record of Bharath Pattabiraman et al. [6], ICA column record algorithm of Golkar Mohammad Javad et al. [14], column RDMC record algorithm of Ricardo C. Correa and associates [35], VNS column records the record of variable neighborhood search algorithm [32]. In some of the above works, if we do not experiment with certain data sets, in the corresponding cells we will record the value -.

3. Evaluation of experimental results

From the experimental results in Table II above, we have the following evaluation comments: With the heuristic

TABLE II
ACHIEVEMENTS ACHIEVEMENTS OF ALTERNATIVES WHEN EXPERIENCE WITH DIMACS SYSTEM

Instance	Heu1	Heu2	GEN	HGAMC	ANT	KLS	PLS	BLS	Heu	ICA	RDMC	VNS
C125.9	34	34	34	-	34	34	-	34	-	34	-	34
C250.9	42	43	44	-	44	44	-	44	-	43	44	44
C500.9	53	54	56	-	56	57	57	57	-	54	-	57
C1000.9	60	64	66	-	67	68	68	68	-	66	-	68
C2000.9	66	62	72	-	73	77	78	80	-	-	-	77
DSJC1000_5	13	14	14	-	15	15	15	-	-	14	15	15
DSJC500_5	13	13	13	-	13	13	-	-	-	13	13	13
C2000.5	14	16	15	-	15	16	16	16	-	-	-	16
C4000.5	16	16	16	-	16	18	18	18	-	-	-	18
MANN_a27	126	125	126	126	126	126	126	126	125	126	126	126
MANN_a45	343	342	343	339	341	345	344	342	341	341	345	345
MANN_a81	1096	1096	1097	-	1092	1100	1098	1094	-	-	-	1096
brock200_2	11	12	12	-	12	11	12	12	10	12	-	12
brock200_4	16	17	16	-	17	16	17	17	14	17	-	17
brock400_2	24	24	24	-	29	25	29	29	20	27	29	29
brock400_4	24	33	25	-	33	25	33	33	22	31	33	33
brock800_2	19	20	20	-	21	21	24	24	18	24	24	24
brock800_4	19	19	20	-	20	21	26	26	17	24	26	26
gen200_p0.9_44	38	42	44	-	44	44	-	44	-	44	44	44
gen200_p0.9_55	43	51	55	-	55	55	-	55	-	54	-	55
gen400_p0.9_55	50	50	55	-	55	53	55	55	-	52	55	55
gen400_p0.9_65	48	50	65	-	65	65	-	65	-	63	65	65
gen400_p0.9_75	52	54	75	-	75	75	-	75	-	71	-	75
hamming10-4	34	33	40	-	40	40	-	40	-	40	-	40
hamming8-4	16	16	16	-	16	16	-	16	16	16	-	16
keller4	11	11	11	11	11	11	-	11	11	11	-	11
keller5	23	27	27	27	27	27	-	27	22	27	-	27
keller6	43	49	55	55	56	57	59	59	45	-	-	59
p_hat300-1	8	8	8	8	8	8	-	8	8	8	-	8
p_hat300-2	25	25	25	-	25	25	-	25	24	25	-	25
p_hat300-3	35	35	36	-	36	36	-	36	26	36	36	36
p_hat700-1	11	11	11	11	11	11	-	11	9	11	-	11
p_hat700-2	44	43	44	-	44	44	-	44	26	44	44	44
p_hat700-3	60	62	62	-	62	62	-	62	-	62	62	62
p_hat1500-1	11	11	11	12	11	12	12	12	-	12	12	12
p_hat1500-2	64	53	65	-	65	65	-	65	-	63	-	65
p_hat1500-3	91	70	94	-	94	94	-	94	-	-	-	94

algorithms Heu1 [31], Heu2 [31] and Heu [6], the Heu2 algorithm is better or equivalent to the Heu algorithm up to 94.1%; Heu2 algorithm is 100.0% better or equivalent to Heu algorithm on 17 data sets where all three of these algorithms have published results.

With algorithms GEN [10], ANT [38], KLS [20], VNS [32], Heu1 [31], Heu2 [31] when experimenting on all 37 datasets in DIMACS [4], the algorithms GEN, ANT, KLS, VNS, Heu1, Heu2 calculations achieved 59.5%, 70.3%, 75.7%, 94.6%, 24.3%, 35.1%, respectively compared with the best known results (ie compared with the Best known column in Table I above).

4. Discuss on running time of algorithms

We also recorded the running times of several algorithms: The running times of the Local search algorithm [15] and the neighborhood search algorithm varied [32] across 10 representative datasets for each data group. as shown in Table III; where the time_LS_MCP column records the time of a run as published [11]; the time_VNS_MCP column records the time of a run as published [32].

The work [32] also tested the dynamic programming algorithm on 37 datasets in the DIMACS system with the program source file published at [49]; in which only 11 data sets give results after 7200 seconds; namely, the datasets C125.9, keller4, brock200_2, brock200_4, gen200_p0.9_55, hamming8-4, p_hat300-1, p_hat300-2, DSJC500_5, p_hat700-1, p_hat1500-1. Finding the right algorithms to solve these data sets is a challenge for current MCP research teams.

IV. CONCLUSION

In this paper, we have presented some empirical investigations on the maximum clique problem, the metaheuristic approaches to solve the maximum clique problem and a standard experimental data system consisting of 37 data sets. We have surveyed and compared the results of some heuristic and metaheuristic algorithms to solve maximum

TABLE III
PROGRAM RUNNING TIME TO PERFORMANCE DATA SET (IN SECONDS)

Instance	time_LS_MCP	time_VNS_MCP
C125.9	5.12	1.34
DSJC1000_5	10.35	5.98
MANN_a27	14.55	20.18
brock400_2	5.03	4.86
gen200_p0.9_44	10.82	3.20
hamming8-4	18.23	1.08
keller4	15.08	0.75
p_hat300-1	3.46	0.61
p_hat700-1	15.36	2.27
p_hat1500-1	123.34	6.42

clique. The heuristic algorithms give acceptable results in fast time, while the metaheuristic algorithms give the solution with the best quality known but much slower time. We also investigated the running time of some algorithms on this standard experimental data system. The results of the other surveys and surveys in this paper can be a source of reference information for further studies to solve the maximum clique problem.

ACKNOWLEDGMENTS

This paper received financial support from the Science and Technology Project of Saigon University: Title: "Surveying some metaheuristic algorithms for solving the maximum clique problem", Subject code: CS2021-11, the group of authors implementing the project includes Dr. Phan Tan Quoc, Master Huynh Thi Châu Ai, Master Do Minh Vu, and Master Truong Minh Hoang Thong.

REFERENCES

[1] Alessio Conte et al, "Finding all maximal cliques in very large social networks", *19th International Conference on Extending Database Technology*, ISBN 978-3-89318-070-7, pp.173-184, Bordeaux, France, 2016.

[2] Andreas Bärmann et al, "The clique problem with multiple-choice constraints under a cycle-free dependency graph", *Discrete Applied Mathematics*, Vol. 283, pp.59-77, ELSEVIER, 2020.

[3] Atul Srivastava et al, "Maximum clique finder: MCF", *IJITEE*, Volume 7, Issue 2, 2018.

[4] Benchmark maximum_clique problem, http://iridia.ulb.ac.be/ fmascia/maximum_clique/DIMACS-benchmark, (last update, 26 Oct 2015).

[5] Benchmark maximum_clique problem, http://sites.nlsde.buaa.edu.cn/ kexu/benchmarks/graph-benchmarks.htm, (last updated: Dec. 1, 2014).

[6] Bharath Pattabiraman et al, "Fast algorithms for the maximum clique problem on massive graphs with applications to overlapping community detection", *Internet Mathematics*, Volume 11, Issue 4-5, 421-448, Taylor & Francis Group, LLC, 2015.

[7] Bo Huang, *"Finding maximum clique with a genetic algorithm"*, The Pennsylvania State University, Master of Science, 2002.

[8] Dam Thanh Phuong, Ngo Manh Tuong, Khoa Thu Hoai, "Maximum clique problem - applications and computational challenges", *Journal of Science & Technology – Issue Natural Science & Engineering, Thai Nguyen University, ISSN: 1859-2171*, Vol.102, No.2, pp.13-17, 2013. (Vietnamese).

[9] Do Minh Vu, *"Bees algorithm to solve the max clique problem"*, Master of Computer Science, Sai Gon University, 2020. (Vietnamese).

[10] Elena Marchiori, "Genetic, Iterated and multistart local search for the maximum clique Problem", *Applications of Evolutionary Computing*, pp.112–121, Springer, 2002.

[11] Emmanuel Hebrard, George Katsirelos, "Conflict directed clause learning for the maximum weighted clique problem", *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp.1316-1323, 2018.

[12] Giannis Nikolentzos et al, "K-clique-graphs for dense subgraph discovery", *Machine Learning and Knowledge Discovery in Database*, pp.617–633, 2017.

[13] George Manoussakis, *Algorithms for cliques in inductive k-independent graphs*, University Paris Sud, France, 2016.

[14] Golkar Mohammad Javad et al, "Maximum clique problem solving using imperialist competitive algorithm and a greedy method for generating initial population", *Indian Journal of Scientific Research*, Vol. 7, Issue 1, pp.442-448, 2014.

[15] Huynh Thanh Tan, Nguyen Van Thanh, "A improved local search for the maximum clique problem",*National Conference, X, Fundamental and Applied Information Technology - FAIR'2017*; Da Nang University, Natural Science and Technology Publishing House. 17-18/08/2017, ISBN:978-604-913-614-6.pp.148-155. (Vietnamese).

[16] Huynh Thi Chau Ai, *"Approximate algorithm to solve the max clique problem"*, Master of Computer Science, Sai Gon University, 2019. (Vietnamese).

[17] Jack J. Dongarra (2014). *Performance of various computers using standard linear equations software*, University of Manchester, pp.1-109 (http://netlib.org/benchmark/performance.pdf).

[18] Jose L. Walteros, Austin Buchanan, "Why is maximum clique often easy in practice?", *Operations Research*, Vol. 68, No. 6, pp.1866–1895, INFORMS, 2020.

[19] José Angel Riveaux Merino, "An exact algorithm for the maximum quasi-clique problem", *International Transactions in Operational Research*, pp.2199-2229, WILEY, 2019.

[20] Kengo Katayama, Akihiro Hamamoto, Hiroyuki Narihisa, "An effective local search for the maximum clique problem", *Information Processing Letters*, Volume 95, Issue 5, pp.503-511, Elsevier, 2005.

[21] Krishna Kumar Singh, Ajeet Kumar Pandey, "Survey of algorithms on maximum clique problem", *IARJSET*, Vol. 2, Issue 2, pp.15-20, 2015.

[22] Melisew Tefera Belachew, Nicolas Gillis, "Solving the maximum clique problem with symmetric rank-one nonnegative matrix approximation", *Journal of Optimization Theory and*

Applications, Vol. 173, pp.279-296, Springer, 2017.
[23] Mochamad Suyudi, Sukono et al, "Branch and bound algorithm for finding the maximum clique problem", *Proceedings of the International Conference on Industrial Engineering and Operations Management*, Bandung, Indonesia, pp.2734-2742, IEOM Society International, 2018.
[24] Mohammad Soleimani-Pouri et al, "Finding a maximum clique using ant colony optimization and particle swarm optimization in social networks", *ASONAM '12: Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pp.58-61, IEEE, 2012.
[25] Muhammad Fayaz et al, "Approximate maximum clique algorithm (amca): a clever technique for solving the maximum clique problem through near optimal algorithm for minimum vertex cover problem", *International Journal of Control and Automation (IJCA)*, Vol. 11, No. 3, pp.35-44, SERSC Australia, 2018.
[26] Nguyen Duc Nghia, Nguyen To Thanh, *"Discrete Math"*, Viet Nam National University Press, HaNoi, pp. 147-260, 2009. (Vietnamese).
[27] Nguyen Canh Nam, "On globally solving the maximum weighted clique problem", *Vietnam National Foundation for Science and Technology Development (NAFOSTED)*, https://optimization-online.org, Viet Nam, 2013.
[28] Phan Tan Quoc, Nguyen Duc Nghia (2013). "Tabu search Algorithm for Solving Minimum Routing Cost Spanning Tree Problem". *Journal on Information Technology & Communication, Ministry of Information and Communications*, ISSN: 1589-3526, Vol. 3, pp.5-13. (Vietnamese).
[29] Phan Tan Quoc, Nguyen Duc Nghia (2013). "Bees algorithm for solving minimum routing cost spanning tree problem". *Journal of Computer Science and Cybernetics*, ISSN: 1813-9663, Vol. 29, No. 3, 2013, pp.265-276, Science and Technics Publishing House. (Vietnamese).
[30] Phan Tan Quoc, "An analysis on the intensification and the diversification properties in metaheuristic algorithms to solve optimization problems", *National conference on information and communication technology*, Can Tho University, ISBN:978-604-919-456-6, 2015. (Vietnamese).
[31] Phan Tan Quoc, Huynh Thi Chau Ai, "Improved heuristic algorithms for solving maximum clique problem", *National Conference, Fundamental and Applied Information Technology) - FAIR'2019*, Institute of Information and Communication Technology; University of Science - Hue University, 07-08/06/2019. Natural Science and Technology Publishing House. ISBN: 978-604-913-915-4, pp.64-72. (Vietnamese).
[32] Phan Tan Quoc, Huynh Thi Chau Ai, Nguyen Son Lam, Huynh Thanh Tan, "A improved variable neighborhood search algorithm to solve the max clique problem", *National Conference, XXII: Some selected issues of Information and Communication Technology, Institute of Information and Communication Technology*. Thai Binh University, 28-29/6/2019. Science and Technics Publishing House, ISBN 978-604-67-1287-9. (Vietnamese).
[33] Qinghua Wua, Jin-Kao Hao, "A review on algorithms for maximum clique problems", *European Journal of Operational Research*, Volume 242, Issue 3, pp.693-709, Elsevier, 2015.
[34] Rachel Behar, Sara Cohen, "Finding all maximal connected s-cliques in social networks", *Proceedings of the 21st International Conference on Extending Database Technology*, ISBN 978-3-89318-078-3, pp.61-72, Vienna, Austria, 2018.
[35] Ricardo C. Correa, Philippe Michelon, Bertrand Le Cun, Thierry Mautor, Diego Delle Donne, "A bit-parallel russian dolls search for a maximum cardinality clique in a graph", https://arxiv.org/pdf/1407.1209.pdf, 2015.
[36] Ryan A. Rossi et al, "Fast maximum clique algorithms for large graphs", WWW '14 Companion: *Proceedings of the 23rd International Conference on World Wide Web, ACM*, pp.365-366, 2014.
[37] Satoshi Shimizu, Kazuaki Yamaguchi,and Sumio Masuda, *A maximum edge-weight clique extraction algorithm based on branch-and-bound*, Kobe University, pp.1-15, 2018.
[38] Serge Fenet, Christine Solnon, "Searching for maximum cliques with ant colony optimization", *Springer-Verlag Berlin Heidelberg*, pp.236-245, 2003.
[39] Shaowei Cai, Jinkun Lin, "Fast solving maximum weight clique problem in massive graphs", *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, pp.568-574, 2016.
[40] Truong Minh Hoang Thong, *"Genetic algorithm to solve the max clique problem"*, Master of Computer Science, Sai Gon University, 2020. (Vietnamese).
[41] Una Benlic, Jin-Kao Hao, "Breakout local search for maximum clique problems", *Computers & Operations Research* Volume 40, Issue 1, pp.192-206, Elsevier, 2013.
[42] Vu Dinh Hoa, Do Trung Kien, "Parallel algorithm to solve the problem of determining the maximum clique on the graph", *National Conference, XII: Some selected issues of Information and Communication Technology*, pp.426-442, 2009. (Vietnamese).
[43] Wayne Pullan, Franco Mascia, Mauro Brunato, "Cooperating local search for the maximum clique problem", *Journal of Heuristics*, Vol. 17, pp.181–199, Springer, 2010.
[44] Yi Zhou, *"Optimization algorithms for clique problems"*, grade de Docteur de l'Université d'Angers, pp.1-122, 2018.
[45] Yuquan Guo et al, "Heuristic artificial bee colony algorithm for uncovering community in complex networks", *Mathematical Problems in Engineering*, pp.1-12, 2017.
[46] Xin-She Yang. *"Engineering optimization: An Introduction with Metaheuristic Applications"*, WILEY, 2010.
[47] Xin-She Yang. *"Nature-inspired metaheuristic algorithms"*. LUNIVER Press, 2010.
[48] http://compbio.ucsd.edu/communities-and-cliques/
[49] https://github.com/bobogei81123/bcw_codebook/blob/master/codes/Graph/Maximum_Clique/Maximum_Clique.cpp
[50] Phan Thanh Huan, Huynh Thi Chau Ai, Chau Le, Sa Lin, "Heuristic Approach for Solving the Maximum Clique Problem on Simple Undirected and Unweighted Graph", *Journal on Information Technology & Communication*, Vol. 2021, No 1, pp.32-39, June, 2021. (Vietnamese).
[51] János Barta, Roberto Montemanni, "The Maximum Clique Problem for Permutation Hamming Graphs", *Journal of Optimization Theory and Applications*, Volume 194, pp.492–507, Springer, 2022.
[52] Wataru Nakasone, Morikazu Nakamura, "Maximal Clique Problem Formulation for Common Structure Detection in Many Graphs", *37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, Phuket, Thailand, IEEE, 2022.

Phan Tan Quoc received Master degree of computer science in 2002 from University of Science, Vietnam National University Ho Chi Minh City and received Doctor degree of computer science in 2015 from Hanoi University of Science and Technology (HUST). He is currently a lecturer of Information Technology Faculty of Sai Gon University, Ho Chi Minh City, Viet Nam. His research interests are metaheuristic algorithms.
Email: quocpt@sgu.edu.vn